初歩からのHDLテストベンチ

第８回　期待値の比較

安岡貴志

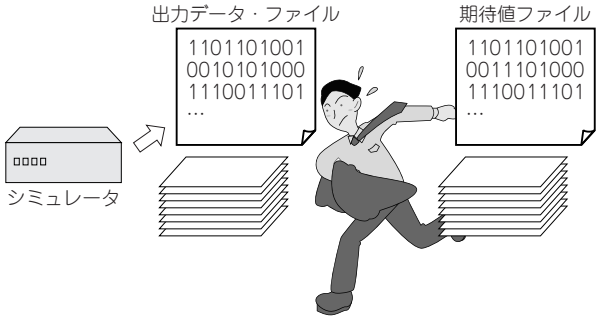
デバイスの記事　ビギナーズ

HDLで回路を記述できるようになったばかりで，これからテストベンチを書こうとしてる方を対象とした連載の第８回である．前回（本誌2008年７月号，pp.113-121の第７回）まででテストベンチに必要な文法の解説は，ほとんど終了した．今回は第５回（本誌2008年３月号，pp.107-115）でも触れた期待値の比較に関して，より作業効率の良い自動化の方法を解説する．　**（筆者）**

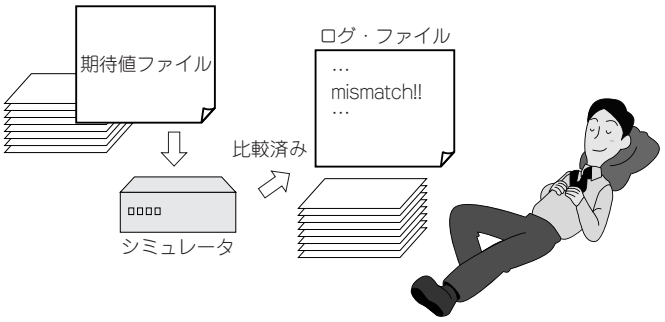
1．期待値の比較を自動化する

Verilog HDL・VHDL共通

本連載の第5回では，期待値ファイルとシミュレーションの結果ファイルを，diffコマンドなどで比較する方法を紹介しました．しかし，テスト・パターン数が増えてくると，すべてのパターンを手動で期待値チェックすることも非常に煩わしくなります．また，人手が入ることでケアレス・ミスを生みやすくなります（**図1**）．



（a）手動比較は大変



（b）自動比較は簡単

図1　出力結果の比較

Verilog HDLやVHDLでは，期待値ファイルさえあれば，テストベンチの中で比較することができます．ここでは，期待値の比較を自動化する方法を説明します．

図2は，クリップ機能付きの加算回路です．この回路は入力A，Bの加算結果をQに出力します．ただし，加算した結果が16以上になった場合には，15を出力します．この回路の名前はaddexとします．

最初にこの回路を検証するテストベンチについて解説します．なお，今回のテストベンチでは期待値不一致のメッセージを標準出力に出しています．ツール側でログ・ファイルを出力する場合はこのままで構いませんが，そうでない場合には標準出力ではなくファイルにメッセージを残すようにしてください（連載第5回を参照）．

Verilog HDL

リスト1は，自動比較機能付きのテストベンチです．

リスト1の①は，期待値ファイルを読み出すための配列

Keyword　シミュレーション・モデル，RAMモデル，include文，package宣言，階層化，ModelSim，ライブラリ

リスト1　自動比較機能付きテストベンチ（Verilog HDL）

```
module addex_tp;
parameter STEP   = 100,
          STROBE = 10,
          PATNUM = 8;
reg  [7:0] mem1 [0:PATNUM-1];
reg  [3:0] mem2 [0:PATNUM-1];
reg  [3:0] a,b;
wire [3:0] q;
integer    i,j;

addex addex( .A(a), .B(b[2:0]), .Q(q) );

initial begin
  $readmemh("input.txt",mem1);
  for(i=0;i<PATNUM;i=i+1)begin
    {a,b} = mem1[i];
    #STEP;
  end
  $finish;
end
```

①期待値データを入れる配列

⑥b[3]は捨てるだけ

```
initial begin
  $readmemh("expect.txt",mem2);
  for(j=0; j<PATNUM; j=j+1) begin

    #(STEP-STROBE) if( mem2[j] !== q )begin
      $display("Mismatch Error!");
      $display("q=%h expect=%h",q,mem2[j]);
    end
    #STROBE;
  end
end
endmodule
```

②期待値読み出し

④期待値比較

③タイミング調整

⑤不一致時のメッセージ

表示されるメッセージ
Mismatch Error!
q= 1 expect= F

回路の出力

期待値

を宣言しています．出力qは4ビットの信号なので，配列のビット幅は4ビットです．期待値データの数は，テスト入力のデータ数と同じでPATNUM（=8）としているので，配列の番地の数は8個（0番地～7番地）です．

リスト1の②は，期待値ファイルexpect.txtを配列mem2に読み出しています．

リスト1の③は，期待値比較するタイミングを調整しています．入力信号は，0，100，200，…，700ns（#1=1nsのとき）で切り替わるようになっていますが，期待値比較は90，190，290，…，790ns（#1=1nsのとき）で行うように調整されています．

リスト1の④は出力信号qと期待値を比較しています．ここで注意すべきことは，比較が!=ではなく，!==だということです（p.129のコラム「等号演算子」を参照）．

リスト1の⑤は，出力信号qと期待値が一致しなかった場合にメッセージを出力します．「Mismatch Error!」の文字とともに，現在のqの値とそれに対する期待値を表示します．2行に分けているのは，単純に見やすさのためです．1行で書いてしまっても問題ありません．

リスト1の⑥のように，b[3]は捨てられています．テスト入力ファイルinput.txtには，**図3（b）**のように，1行2文字でテスト入力が書かれています．この2文字の区切り方として，**図3（c）**のようにポートA，Bに与えるデータを詰めて書いてしまうと，それぞれに与えられるデータが何か，人の目でファイルを見たときに，見にくくなってしまいます．そこで**図3（d）**のように，ポートBに与えるデータを4ビットで表記し，実際に与えるときに1ビット切り捨てています．

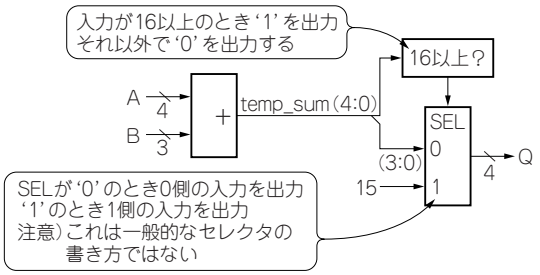


図2　クリップ機能付き加算回路
addexの回路構造を示す．

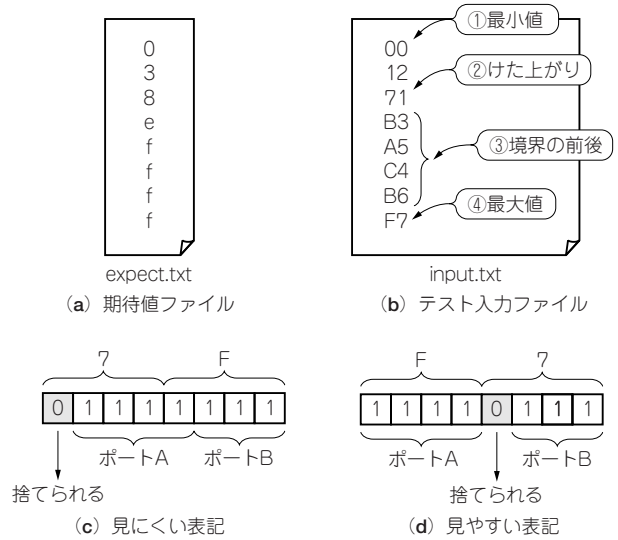


図3　期待値ファイルとテスト入力ファイル
テスト入力ファイルは，見やすい記述を心がける．